H

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/822,090	03/30/2001	Guei-Yuan Luch	42390P10798	9064

8791 7590 08/16/2004

BLAKELY SOKOLOFF TAYLOR & ZAFMAN
12400 WILSHIRE BOULEVARD
SEVENTH FLOOR
LOS ANGELES, CA 90025-1030

EXAMINER	
STEELMAN, MARY J	

ART UNIT	PAPER NUMBER
2122	

DATE MAILED: 08/16/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/822,090	LUEH, GUEI-YUAN	
	Examiner	Art Unit	
	Mary J. Steelman	2122	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *26 April 2004*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-39* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-39* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This action is in response to Amendment A, dated 04/22/2004. Claims 4, 9, 10, 13, 14, 15, 19, 24, 28-30, 34, and 37-39 have been amended. Claims 1-39 are pending.

Drawings

2. In view of the amendment to the Specification, the prior objection to the drawings is hereby withdrawn.

Specification

3. In view of the amendment to the Specification and Applicant's remarks, the prior objections to the Specification are hereby withdrawn.

Claim Rejections - 35 USC § 112

4. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

5. Regarding claims 13-15, 28-30, 35, and 37-39: Referring to the statement, "A trademark or trade name is used to identify a source of goods, and not the goods themselves", the trademark JAVA identifies the source, i.e., Sun Microsystems, Inc., not the goods themselves. Applicant could modify the claims through the use of additional generic terminology, e.g., 'JAVA programming language method / virtual machine / virtual machine debug interface', in order to properly describe the instant invention.

6. Lack of Antecedent basis:

Claim 35 recites the limitation "the stub" in line 3. There is insufficient antecedent basis for this limitation in the claim. Changing "the stub" to "a stub" could cure this.

Claim Rejections - 35 USC § 103

7. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

8. Claims 1, 4-8, 13-16, 19-23, 28-31, 34, and 37-39 are rejected under 35 U.S.C. 103(a) as being unpatentable over US Patent 6,078,744 to Wolczko et al., in view of US Pre Grant Publication 2001/0047510 A1 to Angel et al.

Per claims 1, 16 and 31:

Wolczko disclosed re-compiling bytecode into native code (col. 6, 51-54, "...recompiles portions of the source program and generates optimized machine code..." using a JIT compiler (col. 6, line 36), when changing portions of a program during optimization. Abstract, lines 1-3, "Apparatus (a system as noted in claim 31), methods (as noted in claim 1), and computer program products (as noted in claim 16) are disclosed for improving the performance of subsequent compilations of a source program.

Wolczko failed to disclose:

-when a field watch for a field is activated, the function including a byte code sequence having a field byte code that accesses or modifies the field;

-generating an instrumentation code corresponding to the field watch of the field;

-inserting the instrumentation code to the native code.

However, Angel, US Pre Grant Publication 2001/0047510 A1 disclosed [0184], "the monitoring DLL in connection with providing data to the analyzer/viewers...may be implemented as native code that is called via the monitoring function calls that are inserted into the instrumented class..." and [0015], "...method parameter may be passed...from an instrumentation runtime function...", [0052], "Code instrumentation software...accesses the transitional representations and adds instrumentation instructions (inserting instrumentation code)...", [0091], "it is useful to instrument memory access instructions (fields)...monitoring (field watch activated) the variables (fields) of a program that access (accesses or modifies field) memory...", [0111], "...instructions being instrumented relate to memory variable (byte code that accesses field) accesses..." The 'field' could be the means of determination as to whether a node in the IR tree is a 'node of interest'. If the field is activated, the node is chosen for instrumentation [0089] (function including a byte code sequence having a field byte code that accesses or modifies the field.) Angel [0123], "Once the instrumented IR data element is provided, then...the compiler may continue the compile process (recompile)..." Any memory read / write instruction is able to access / modify a field. Angel disclosed [0016], "A native function call may be instrumented by adding a byte code wrapper to the native function and then instrumenting the wrapper (inserting the instrumentation code to the native code)." [0090]: "...if it is determined...that the CN is a node of interest, then control passes...to a step where a portion of the IR tree is instrumented, either by replacing the CN and/or adding additional nodes (generating an instrumentation code corresponding to the field watch)..." [0091]: "...it is useful

to instrument memory access instructions (access or modify)...", [0125], "...automatically editing the executable byte code representation of...methods for generating instrumented byte code." (emphasis added), [0127], "One objective of the instrumentation process is to alter the program to facilitate the gathering of diagnostic (field watch activated)...information on the program when it is executed..." Angel provided references that determine if a field is to be 'watched', then instrumentation code is generated and inserted.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention to have modified Wolczko's invention that recompiles altered code dynamically, when optimizing and changing a small portion of the program (col. 1, lines 41-42), by including the features disclosed by Angel, altering code with instrumentation, recompiling as necessary, and otherwise using previously compiled portions, thereby reducing a startup delay.

Per claims 4, 19, and 34:

Wolczko disclosed recompiling portions of program code when making changes during optimization. Angel provided more details regarding the alteration of the code prior to recompiling. Wolczko failed to provide information related to saving state, executing and event hook then restoring state.

However, Angel disclosed:

-saving live global state, the live global state corresponding to an active register; ([0169], "...routine is then patched...at runtime, each call..is intercepted...", [0170], "The patch uses an assembly code thunk that includes a small amount of assembly code and a class instance (data structure) that lets the patch code get control (this is done by saving state) before the native code

routine starts...", [0174], "The assembly thunk code may put a pointer...into whichever register (corresponding active register)...")

-executing an event hook function for an event corresponding to the field watch; ([0170], "patch code get control before the native code routine starts, and also gets control back when the native code routine exits." (code is executed), also [0137], "The instrumentation program operates in cooperation with the VM runtime system and may take advantage of particular hooks...")

-restoring the live global state. ([0170], "and also gets control back (restore state from registers) when the native code routine exits.")

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to have modified Wolczko's invention that recompiles altered code dynamically, when optimizing and changing a small portion of the program (col. 1, lines 41-42), by including features as disclosed by Angel: saving state, executing a function, and restoring state, as this is well known in the art.

Per claims 5 and 20, Angel disclosed:

-pushing the live global state onto a stack. ([0153], "parameters that are passed during instrumentation are passed in a conventional fashion using the stack. Thus, the parameters are pushed on to the stack prior to invocation of the monitoring function being called.")

Per claims 6 and 21, Angel disclosed:

-passing an argument corresponding to the field; ([0093], "pass a variable pointer (passing an argument) to a function and have that pointer be assigned to another variable within the

function...", [0114], "The run time instrumentation node may be a function call to a run time instrumentation function that uses the child node as one of the arguments and returns the value of the child node from the function call to make the value available for the operation node.")

-calling a run-time library function related to the event. ([0113], "each of the specific run time instrumentation routines that is provided may include a function that is called to perform the instrumentation operation...", [0119], "The run time instrumentation code may be implemented by using a separate set of routines that is linkable to the code being instrumented via the function calls...The initialization routine determines if an executable library corresponding to the run time instrumentation routine is available...")

Per claims 7 and 22:

-retrieving the live global state from the stack. (See arguments presented in the rejection of claim 4.)

Per claims 8 and 23, Angel disclosed:

-inserting the instrumentation code in a stub at end of the code space. ([0149], "...exit point is instrumented.", [0157], "instrumentation code is inserted at the end of the method...")

Per claims 13, 28, and 37, Angel disclosed:

-the function is a JAVA method. ([0014], "instrumenting a byte code (JAVA) computer program...", [0144], "entry of method is instrumented.")

Per claims 14, 29, and 38, Angel disclosed:

-the field is a JAVA field in a JAVA virtual machine. ([0014], "instrumenting a byte code (JAVA) computer program…", [0147], "byte code is inserted into the method to cause a local line number variable (field) to be set to the new line number when the method runs.")

Per claims 15, 30 and 39, Angel disclosed:

-the event hook function is compatible with a JAVA Virtual Machine Debug Interface (JVMDI). ([0137], "Instrumentation program operates in cooperation with the VM runtime system and may take advantage of particular hooks (a virtual machine debug interface) or calls provided by the vendors of the VM runtime system." A JAVA Virtual Machine Debug Interface, JVMDI, is a debug interface, that is trademarked by Sun Microsystems.)

9. Claims 2, 3, 9-12, 17, 18, 24-27, 32, 33, 35 and 36 are rejected under 35 U.S.C. 103(a) as being unpatentable over US Patent 6,078,744 to Wolczko et al., in view of US Pre Grant Publication 2001/0047510 A1 To Angel et al., and further in view of "Poor Man's Watchpoints", by Max Copperman and Jeff Thomas (1995).

Per claims 2, 3, 17, 18, 32 and 33:

Wolczko disclosed recompiling portions of program code when making changes during optimization. Angel provided more details regarding the alteration of the code prior to recompiling. Neither Wolczko nor Angel disclosed enabling / disabling the execution of instrumented code.

However, Copperman and Thomas disclosed:

-guarding execution of the instrumentation code if the field watch is not activated; executing a filed watch sequence if the field watch is activated. (Page 38, The Debuggee, 3rd paragraph, "When no watchpoints are set..." (disabled/guarded), page 40, maintaining the Watch Table, 2nd paragraph, "When a watchpoint command is entered or enabled (activated)...When a command is disabled or canceled (guarded / not activated) ...")

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to have modified Wolczko and Angel, by using Copperman and Thomas's disclosure that allows a user to control the enabling / disabling of watchpoints when inserting code modification, thereby making optimization more useful.

Per claims 9, 24, and 35, Angel disclosed:

-updating an offset of a jump instruction to the stub when the field watch is activated. ([0176], "The records contain the new offset of the byte code instructions, which are moved due to insertion of instrumentation instructions.", [0181], "...the code table...to reflect the new offsets of the instrumented byte code...", [0182], (...byte code is modified to update branch instructions to reflect the new offsets...")

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to have modified Wolczko's invention that recompiles portions of byte code into native code when altering small portions of code during optimization, by including details provided by Angel regarding the necessary change in offsets due to the addition of instrumentation code.

Per claims 10, 25, and 36:

Wolczko disclosed recompiling portions of program code when making changes during optimization. Angel provided more details regarding the alteration of the code prior to recompiling. Neither Wolczko nor Angel disclosed enabling code with a jump instruction.

However, Copperman and Thomas disclosed:

-replacing a no-op sequence with a jump instruction to the stub. (Page 37, Introduction, 4th paragraph, "code patching - replacing each store and/or load instructions with an inline check or call to a function that gives control to the debugger if the accessed location is being watched, and subsequently executes..." Also, page 40, The Debugger, 3rd paragraph, "On receiving a watchpoint command, the debugger has to add an entry to the watch table and ensure that <cmd> is executed (jump to stub / instrumentation code) when the watchpoint is hit. Also, page 40, last paragraph, "user's command must be executed at the patch target (stub / instrumented code).")

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to have modified Wolczko's and Angel's invention by including information as provided by Copperman and Thomas regarding jumping to an inserted patch / instrumented code portion / stub for the purpose of optimizing through instrumentation, when altering control flow of a program as these are well known techniques.

Per claims 11, 12, 26 and 27:

Wolczko disclosed recompiling portions of program code when making changes during optimization. Angel provided more details regarding the alteration of the code prior to

recompiling. Neither Wolczko nor Angel disclosed clearing code that would otherwise execute an instrumented portion.

However, Copperman and Thomas disclosed:

-clearing the field watch by replacing the offset with a zero offset. (Page 38, The Dubuggee, 3r^{d} paragraph, "When no watchpoints are set...made the first instruction in the patch branch around the rest of the patch if $fp contains zero...")

-clearing the field watch by replacing the jump instruction with the no-op sequence. (Page 40, Maintaining The Watch Table, 2nd paragraph, "When a command is disabled or canceled, the last range in the table is copied over the range that is no longer being watched...If the table is empty, $fp is set to zero..., page 41, 3rd paragraph, "...code to...disable, enable and cancel individual watchpoints...")

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to have modified Wolczko and Angel, by using Copperman and Thomas's disclosure that provides more information regarding clearing the watch field when instrumenting code because these features allow interactive user control, thereby making optimization through instrumentation more flexible.

Response to Arguments

10. Applicant's arguments filed 22 April 2004 have been fully considered but they are not persuasive.

(A) Applicant has argued, in substance, the following:

As noted on page 15, 5th paragraph, of Amendment A, dated 22 April 2004, Applicant has argued, "Wolczko, Angel and Copperman, taken alone or in any combination, does not disclose, suggest, or render obvious (1) re-compiling when a field watch is activated, (2) generating an instrumentation code corresponding to the field watch; and (3) inserting the instrumentation code to the native code.

Examiner's Response:

See detailed response to claim 1 above. The Wolczko reference suggests recompiling altered code. The term 'field watch' is a broad term. The 'field' could be the means of determination as to whether a node in the IR tree is a 'node of interest'. If the field is activated, the node is chosen for instrumentation [0089]. The Angel reference provides a suggestion for code altered by instrumentation to be recompiled. Angel [0123], "Once the instrumented IR data element is provided, then...the compiler may continue the compile process (recompile)..." Any memory read / write instruction is able to access / modify a field. Angel disclosed [0016], "A native function call may be instrumented by adding a byte code wrapper to the native function and then instrumenting the wrapper (inserting the instrumentation code to the native code)." [0090]: "...if it is determined...that the CN is a node of interest, then control passes...to a step where a portion of the IR tree is instrumented, either by replacing the CN and/or adding additional nodes (generating an instrumentation code corresponding to the field watch)..." [0091]: "...it is useful to instrument memory access instructions (access or modify)..."

Thus Examiner maintains that, at least, the combination of Wolczko and Angel disclose the limitations of independent claims 1, 16, and 31.

(B) Applicant has argued, in substance, the following:

As noted on page 15, 5th paragraph, of Amendment A, dated 22 April 2004, Applicant has argued, "there is no motivation to combine Wolczko, Angel and Copperman because none of them addresses the problem of recompilation according to a field watch."

Examiner's Response:

In response to applicant's argument that there is no suggestion to combine the references, the examiner recognizes that obviousness can only be established by combining or modifying the teachings of the prior art to produce the claimed invention where there is some teaching, suggestion, or motivation to do so found either in the references themselves or in the knowledge generally available to one of ordinary skill in the art. See *In re Fine*, 837 F.2d 1071, 5 USPQ2d 1596 (Fed. Cir. 1988) and *In re Jones*, 958 F.2d 347, 21 USPQ2d 1941 (Fed. Cir. 1992). In this case, each reference discloses instrumented code. Wolczko and Angel both disclosed recompilation when attempting to improve performance. The Cooperman reference was combined to include the limitations of disabling / or activating watchpoints when instrumenting a program.

(C) Applicant has argued, in substance, the following:

As noted on page 16, 1st paragraph, of Amendment A, dated 22 April 2004, Applicant has argued, "Wolczko, read as a whole, does not suggest the desirability of generating an instrumentation code corresponding to the field watch."

Examiner's Response:

The Angel reference is combined to reject this limitation.

(D) Applicant has argued, in substance, the following:

As noted on page 16, 2nd paragraph, of Amendment A, dated 22 April 2004, Applicant has argued, "Wolczko does not disclose recompiling a function when a field watch for a field is activated."

Examiner's Response:

The Angel reference is combined to reject this limitation

(E) Applicant has argued, in substance, the following:

As noted on page 16, 3rd paragraph, of Amendment A, dated 22 April 2004, Applicant has argued, "Monitoring memory access instructions and/or variables (as disclosed by Angel) is not the same as activating a field watch of a field."

Examiner's Response:

Angel detects whether the node in the IR tree is a 'node of interest' (detect a field value). If so, an instrumented code may be added. It is useful to monitor memory accesses as they are a common source of program error.

(F) Applicant has argued, in substance, the following:

As noted on page 16, 4th paragraph, of Amendment A, dated 22 April 2004, Applicant has argued that the Copperman reference is not applicable to the "field watch as discussed above."

Examiner's Response:

Copperman instruments a program to check memory accesses using watchpoints (Abstract).

Conclusion

11. THIS ACTION IS MADE FINAL. Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

11. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

12. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Mary Steelman, whose telephone number is (703) 305-4564. The examiner can normally be reached Monday through Thursday, from 7:00 A.M. to 5:30 P.M. If

attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Dam can be reached on (703) 305-4552.

The fax phone number is (703) 872-9306 for regular communications and for After Final communications. Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is (703) 305-3900.

Mary Steelman

08/05/2004

TUAN DAM
SUPERVISORY PATENT EXAMINER